IN THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the present application:

1-29. (New) (Canceled)

30. (New) A method comprising:

maintaining a file system that includes a live filespace accessible to users and a zombie filespace not accessible to users;

in response to a request to delete at least a portion of a file stored in the live filespace, moving blocks storing said portion of the file from the live filespace to the zombie filespace and deleting at least some of the blocks from the zombie filespace before a scheduled consistency checkpoint of the file system;

recording said moving and deleting in a persistent memory; and

in response to a crash of the file system, replaying said moving and deleting based on information stored in the persistent memory to recover the file system.

31. (New) The method of claim 30, wherein the zombie filespace is a portion of the file system where files are not available to users in normal operation, but can still be manipulated by the file system as if the files were normal files.

32. (New) The method of claim 30 further comprising deleting rest of the blocks from the zombie filespace after the scheduled consistency checkpoint of the file system and recording deleting the rest of the blocks in the persistent memory.

33. (New) The method of claim 30, wherein moving blocks storing said portion of the file from the live filespace to the zombie filespace comprises breaking links associating these blocks with the live filespace and creating links associating these blocks with the zombie filespace.

34. (New) The method of claim 30, wherein moving blocks storing said portion of the file from the live filespace to the zombie filespace comprises creating an evil twin file in the zombie filespace and adding these blocks as part of the evil twin file.

35. (New) The method of claim 30, wherein recording said moving and deleting in a persistent memory comprises recording changes of the live filespace and the zombie file space caused by said moving and deleting in the persistent memory.

36. (New) The method of claim 35, wherein replaying said moving and deleting based on information stored in the persistent memory comprises replaying the changes of the live filespace and the zombie file space recorded in the persistent memory.

37. (New) The method of claim 30, wherein the request to delete at least a portion of a file stored in the live filespace comprises a request to delete the file from the live filespace or to truncate the file.

38. (New) The method of claim 30, wherein deleting a block from the zombie filespace comprises returning the block as a free block to the file system.

39. (New) The method of claim 30 further comprising increasing the size of the zombie filespace before moving the blocks.

40. (New) The method of claim 30, wherein the request to delete at least a portion of the file consumes extra long period of time.

41. (New) A method comprising:

maintaining a file system that includes a live filespace accessible to users and a zombie filespace not accessible to users;

in response to a request to delete a file having attached data elements from the live filespace, moving blocks storing the file and the attached data elements from the live filespace to the zombie filespace, and deleting at least some of the blocks from the zombie filespace before a scheduled consistency checkpoint of the file system;

recording said moving and deleting in a persistent memory; and

in response to a crash of the file system, replaying said moving and deleting based on information stored in the persistent memory to recover the file system.

42. (New) The method of claim 41, wherein the zombie filespace is a portion of the file system where files are not available to users in normal operation, but can still be manipulated by the file system as if the files were normal files.

43. (New) The method of claim 41 further comprising deleting rest of the blocks from the zombie filespace after the scheduled consistency checkpoint of the file system and recording deleting the rest of the blocks in the persistent memory.

44. (New) A machine-readable medium having instructions, when executed, cause a machine to perform a method, the method comprising:

maintaining a file system that includes a live filespace accessible to users and a zombie filespace not accessible to users;

in response to a request to delete a file from the live filespace, moving blocks storing the file from the live filespace to the zombie filespace and deleting at least some of the blocks from the zombie filespace before a scheduled consistency checkpoint of the file system;

recording said moving and deleting in a persistent memory; and

in response to a crash of the file system, replaying said moving and deleting based on information stored in the persistent memory to recover the file system.

45. (New) The machine-readable medium of claim 44, wherein the method further comprises deleting rest of the blocks from the zombie filespace after the scheduled consistency checkpoint of the file system and recording deleting the rest of the blocks in the persistent memory.

46. (New) The machine-readable medium of claim 44, wherein moving blocks storing the file from the live filespace to the zombie filespace comprises breaking links associating these blocks with the live filespace and creating links associating these blocks with the zombie filespace.

47. (New) The machine-readable medium of claim 44, wherein recording said moving and deleting in a persistent memory comprises recording changes of the live filespace and the zombie file space caused by said moving and deleting in the persistent memory.

48. (New) The machine-readable medium of claim 44, wherein replaying said moving and deleting based on information stored in the persistent memory comprises replaying the changes of the live filespace and the zombie file space recorded in the persistent memory.

49. (New) A machine-readable medium having instructions, when executed, cause a machine to perform a method, the method comprising:

maintaining a file system that includes a live filespace accessible to users and a zombie filespace not accessible to users;

in response to a request to truncate a file stored in the live filespace, moving blocks storing a portion of the file to be truncated from the live filespace to the zombie filespace and deleting at least some of the blocks from the zombie filespace before a scheduled consistency checkpoint of the file system;

recording changes of the live filespace and the zombie filespace caused by said moving and deleting in a persistent memory; and

in response to a crash of the file system, replaying the changes of the live filespace and the zombie filespace recorded in the persistent memory to recover the file system.

50. (New) The machine-readable medium of claim 49, wherein moving blocks storing a portion of the file to be truncated from the live filespace to the zombie filespace comprises creating an evil twin file in the zombie filespace and adding these blocks as part of the evil twin file.

51. (New) The machine-readable medium of claim 49, wherein the method further comprises deleting rest of the blocks from the zombie filespace after the scheduled consistency checkpoint of the file system and recording changes of the zombie filespace caused by deleting the rest of the blocks in the persistent memory.

52. (New) The machine-readable medium of claim 49, wherein the method further comprises increasing the size of the zombie filespace before moving the blocks.

53. (New) The machine-readable medium of claim 49, wherein the request to truncate the file consumes extra long period of time.

54. (New) A storage server comprising:

a processor;

a network interface through which to communicate with a client;

a storage interface through which to access a plurality of mass storage devices on behalf of the client; and

a memory coupled to the processor, the memory storing instructions which, when executed by the processor, cause the storage server to perform a process comprising:

maintaining a file system that includes a live filespace accessible to the client and a zombie filespace not accessible to the client;

in response to a request to delete at least a portion of a file stored in the live filespace, moving blocks storing said portion of the file from the live filespace to the zombie filespace and deleting at least some of the blocks from the zombie filespace before a scheduled consistency checkpoint of the file system;

recording said moving and deleting in a persistent memory; and

in response to a crash of the file system, replaying said moving and deleting based on information stored in the persistent memory to recover the file system.

55. (New) The storage server of claim 54, wherein the process further comprises deleting rest of the blocks from the zombie filespace after the scheduled consistency checkpoint of the file system and recording deleting the rest of the blocks in the persistent memory.

56. (New) The storage server of claim 54, wherein moving blocks storing said portion of the file from the live filespace to the zombie filespace comprises breaking links associating these blocks with the live filespace and creating links associating these blocks with the zombie filespace.

57. (New) The storage server of claim 54, wherein recording said moving and deleting in a persistent memory comprises recording changes of the live filespace and the zombie file space caused by said moving and deleting in the persistent memory.

58. (New) The storage server of claim 57, wherein replaying said moving and deleting based on information stored in the persistent memory comprises replaying the changes of the live filespace and the zombie file space recorded in the persistent memory.

59. (New) The storage server of claim 54, wherein the request to delete at least a portion of a file stored in the live filespace comprises a request to delete the file from the live filespace or to truncate the file.

60. (New) The storage server of claim 54, wherein deleting a block from the zombie filespace comprises returning the block as a free block to the file system.

61. (New) The storage server of claim 54, wherein the process further comprises increasing the size of the zombie filespace before moving the blocks.